(2)

SEA TECH INC. FAX 503-757-7027 TELEX 258519 CTEK
P.O. Box 779 • Corvallis, Oregon 97339 • 503-757-9716

PROGRESS REPORT
TO
OFFICE OF NAVAL RESEARCH

FOR CONTRACT NO: N00014-90-C-0123

TITLE: Development of an Expendable Particle Sensor

ITEM NO: 0001AA

DATE: 30 April 92

Robert Bartz
Principal Investigator

Statement A per telecon
Dr. Richard Spinrad ONR/Code 1123
Arlington, VA 22217-5000

NWW 6/10/92

PROGRESS REPORT: Development of an Expendable Particle Sensor

Sea Tech Inc.

Contract No. N00014-90-C-0123

Item No. 0001AA

INTRODUCTION:

This report documents progress related to the Phase II Development of the Expendable Particle Sensor, (EPS) during the time period of August through September 1991. Between the completion of Phase I and the beginning of Phase II funding, considerable progress was made by Sea Tech to improve the design of the EPS. This progress is included in this report to show the status of the EPS development prior to ONR Phase II funding which started in August 1991.

BACKGROUND:

ONR funded a SBIR Phase I program to determine the feasibility of the EPS and this was completed in October 1990. The work carried out during this program was the investigation and testing of a variety of forward and back scattering sensors and the preliminary design and testing of a data telemetry system. This work demonstrated the feasibility of building sensitive, low cost expendable scattering and temperature sensors. Phase II was funded by ONR with a start date of August 1992.

In the interim period between Phase I and Phase II, the scattering sensor development was funded by Sea Tech. During this period considerable improvements were made in the design of the sensor. Scattering sensor designs were developed which could easily be modified to fit in the side of a standard expendable probe, the sensor was designed using dual LEDs to double the light output without using additional power, and tests were done to determine the full range of sediment concentrations measurable with the sensor. In addition, the data telemetry system was redesigned to send and receive five channels of data 12 times per second. This work, funded by Sea Tech, is described in detail in Section 1 of this report.

The continuation of the scattering sensor development, funded by the Phase II ONR SBIR contract shown above became effective on 21 August 1992. Section 2 of this report documents progress related to the development of the expendable underwater forward scattering sensor from August 21 to September 30, 1991. Work during this time consisted primarily of breadboarding and testing the five-channel data telemetry system. A detailed account of this work is presented in section 2 of this report.

RESULTS:

SECTION 1, WORK FUNDED BY SEA TECH, OCTOBER 1990 TO AUGUST 1991

As stated in the introduction several improvements were made in the scattering sensor design between the completion of Phase I and the beginning of Phase II. A non-expendable underwater forward scattering sensor micro structure probe was built and tested in Oregon State University's Chameleon microstructure instrumentation package. The design was subsequently simplified and is better suited for use with standard expendable probes to the extent that only very minor modifications to the probe and the sensor package will be necessary to install the sensor in the

Sparton expendable probe body. The drawing for this probe is shown in Figure 1 and described in Appendix A.

The light output of the scattering sensor was doubled using the same power by connecting two LEDs in series; the old design with a single LED did not take full advantage of the available voltage.

Several sensor designs were tested to determine the high sediment concentration end of the linear range of the scattering sensor. Figure 2 is a plot of the results; with the light source close to the detector, the scattering sensor responds linearly up to sediment concentrations of several grams per liter. In Phase I it was shown that the forward light scattering sensor was capable of measuring concentrations as low as a few micrograms per liter, as shown in Figure 3. These two figures show that the linear range of this sensor extends from a few micrograms per liter to greater than a few grams per liter, exceeding six orders of magnitude of sediment concentrations. Thus the forward light scattering sensor should be capable of measuring any naturally occurring sediment concentration in either ocean or fresh water. The performance of the forward light scattering sensor for high concentrations of suspended sediment is discussed in detail in Appendix A.

Work continued with the development of a five-channel telemetry system from the basic system developed in Phase I. The transmission of five channels of data twelve times per second is required to sample the water column with a depth resolution of 1/2 meter assuming a probe drop rate of 6 meters per second. The five channels are: 1. Water Temperature 2. Light Scattering 3. Probe Body Temperature 4. Positive Supply Voltage 5. Ground.

A schematic of the five-channel telemetry system transmitter is shown in Figure 4. The timing diagram, Figure 5, illustrates how this system operates. Three of the NAND gates of the 4011 provide a 12.2 KHz signal which are fed into the 4020 Binary counter which controls the timing for this system and provides 768 Hz to the scattering sensor for modulating the LEDs. Input to the 4051 multiplexer is such that scattering and water temperature voltages are output twice every 167 milliseconds, or 12 times per second (see Figure 5). Body temperature is output 6 times per second, which gives us more than adequate data for temperature compensation of the scattering sensor. We submultiplex the positive supply voltage and ground so that each is transmitted 3 times per second; Q12 of the 4020 switches between the positive supply voltage and ground at 3 Hz.

The output of the 4051 is fed into the 2209 Voltage Controlled Oscillator (VCO) which gives us 500 to 1500 Hz signals for 0 to 5 Volts input. An analog switch, 1/3 of the 4053, operates as a sync gate, interrupting the VCO output every 83 milliseconds with a sync pulse. The sync pulses allow us to synchronize the data receiver with software. The output of the sync gate is fed into a 358 dual operational amplifier which differentially drives the wire data link.

The light scattering, water temperature, and body temperature output voltages are all linear functions of the supply voltage. The relationship between the VCO output frequency and input voltage is also linear. Thus, if the scattering meter is calibrated, turbidity or sediment concentrations can be computed directly from scattering frequency, positive supply frequency, and ground frequency. Similarly, water temperature and probe body temperature can be computed. The probe body temperature can be used to correct for errors caused by the temperature coefficient of the scattering sensor.

The Phase I design of the telemetry receiver hardware did not require changing for the five-channel system. The schematics for this hardware are shown in Figures 6, 7, and 8. The only change we made at this time was removing the Frame Sync, which we found unnecessary. Preliminary software for the five-channel system was written at this time; the program NEWEXP.PAS, written in Turbo Pascal, is found in Appendix C.

SECTION 2, WORK FUNDED BY ONR, AUGUST AND SEPTEMBER 1991

During the first two months of Phase II, we breadboarded the five-channel telemetry system and ran some preliminary tests. The frequency response of the equalization network was measured and is shown in Figure 9. The peak response is in the middle of the 500-1500 Hz band over which we transmit data. Noise outside this band is effectively filtered out.

We sent and received five channels of data through the telemetry system at the designed rates (12 times per second for the water temperature and light scattering channels). Our inputs into the scattering, water temperature, and probe body temperature channels of the telemetry transmitter were known voltages. Figure 10 shows that were able to accurately determine the input voltages from the received frequencies for voltages between 0 and 6 volts.

The noise of the breadboarded telemetry system can also be seen in Figure 10. It is about 3 millivolts for 6 Volts full scale, or 0.03%. This gave us an early idea of what to expect for our noise floor, although this was neither a best case nor a worst case test. On the one hand we expect that the noise level of the telemetry system itself will go down when we lay out the circuitry carefully on a printed circuit board, but on the other hand we expect the scattering sensor, with its 750 Hz switching of 50 mA current to cause ground loop problems, and possibly raise the noise floor.

FIGURES 1 to 10

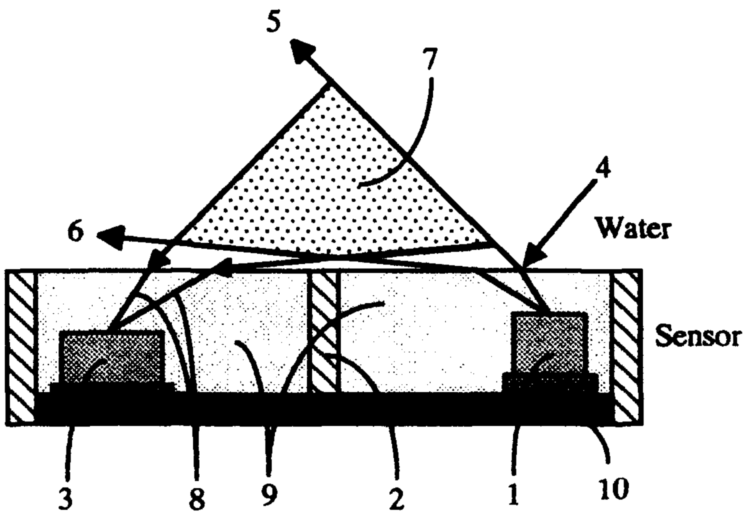


Figure 1 Underwater Forward Light Scattering Sensor

Figure 2 Calibration of Scattering Sensor with Clay at Low Gain for Extended Range

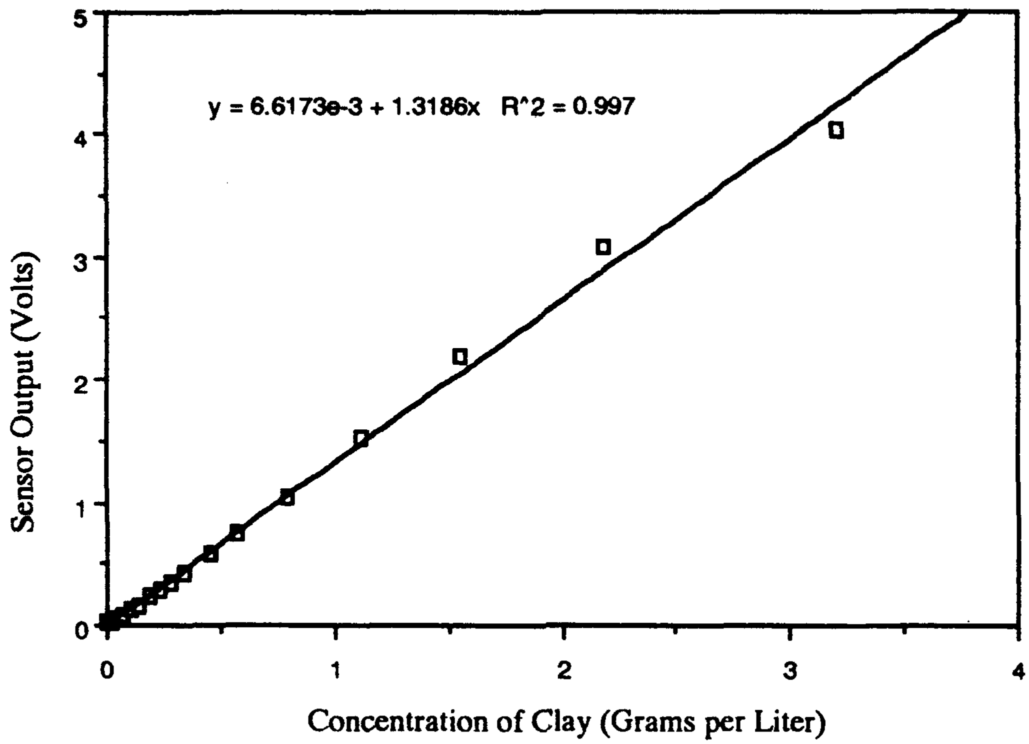
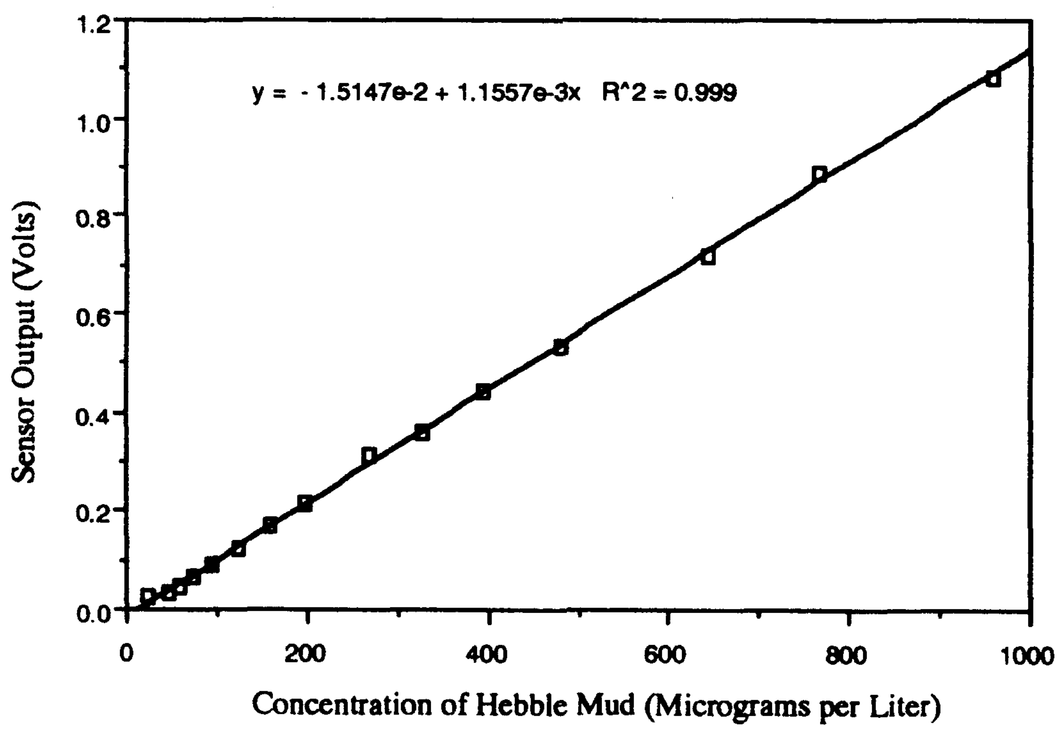


Figure 3 Calibration of Scattering Sensor with Hebble Mud at High Gain for High Sensitivity



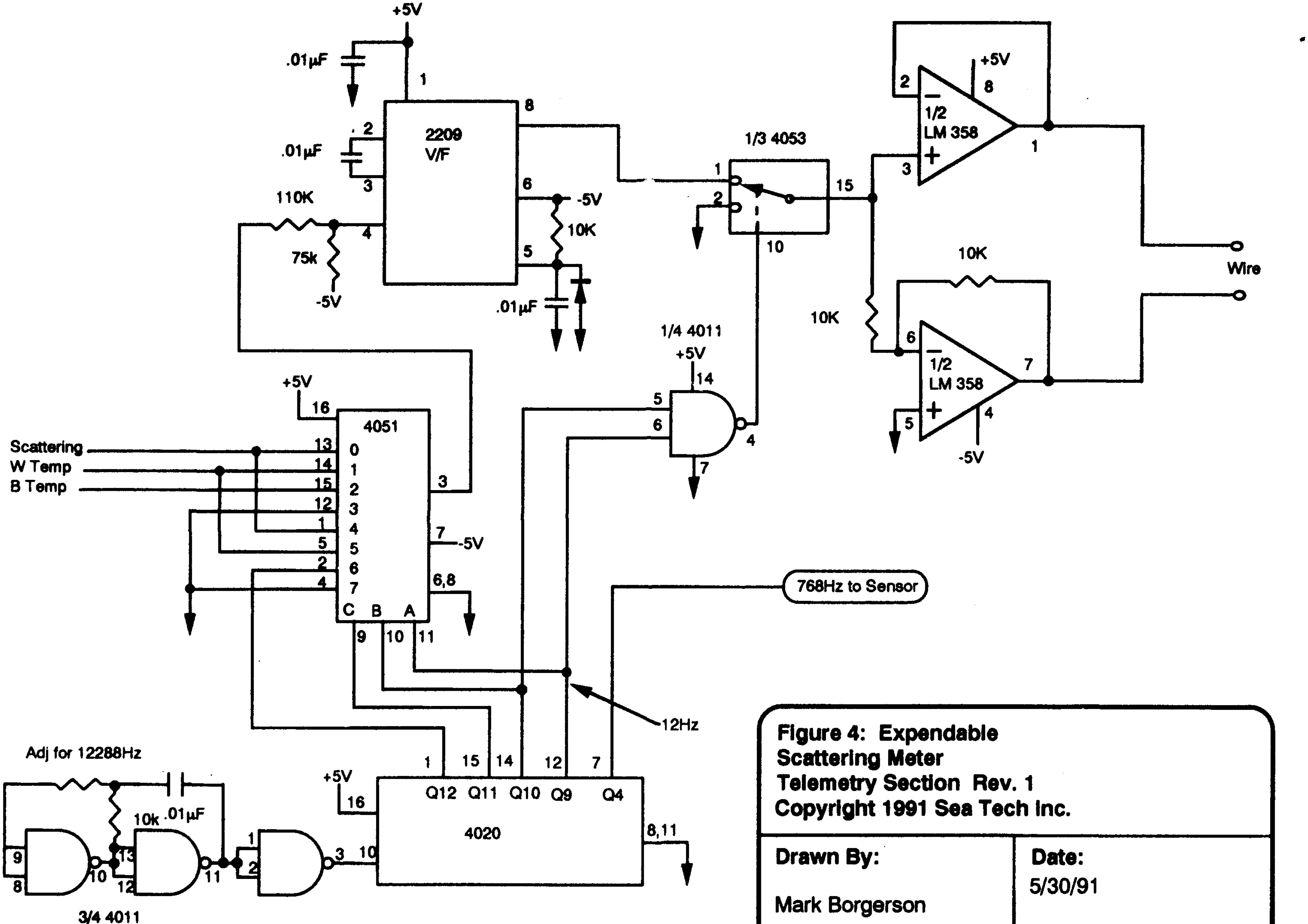


Figure 4: Expendable Scattering Meter Telemetry Section Rev. 1
Copyright 1991 Sea Tech Inc.

Drawn By: Mark Borgerson

Date: 5/30/91

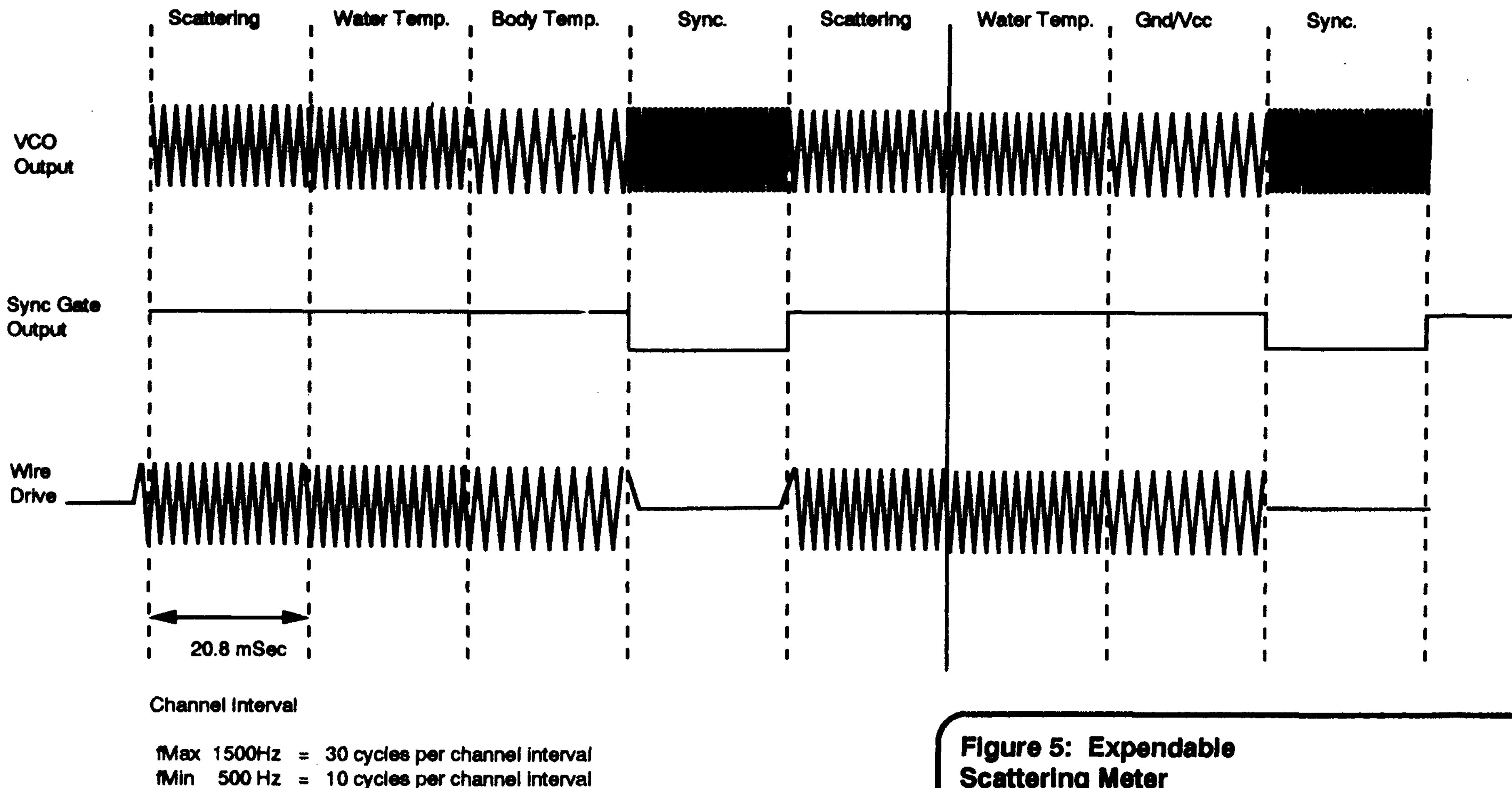


Figure 5: Expendable Scattering Meter Telemetry Output Waveform Copyright 1991 Sea Tech Inc.	
Drawn By: Mark Borgerson	**Date:** 5/30/91

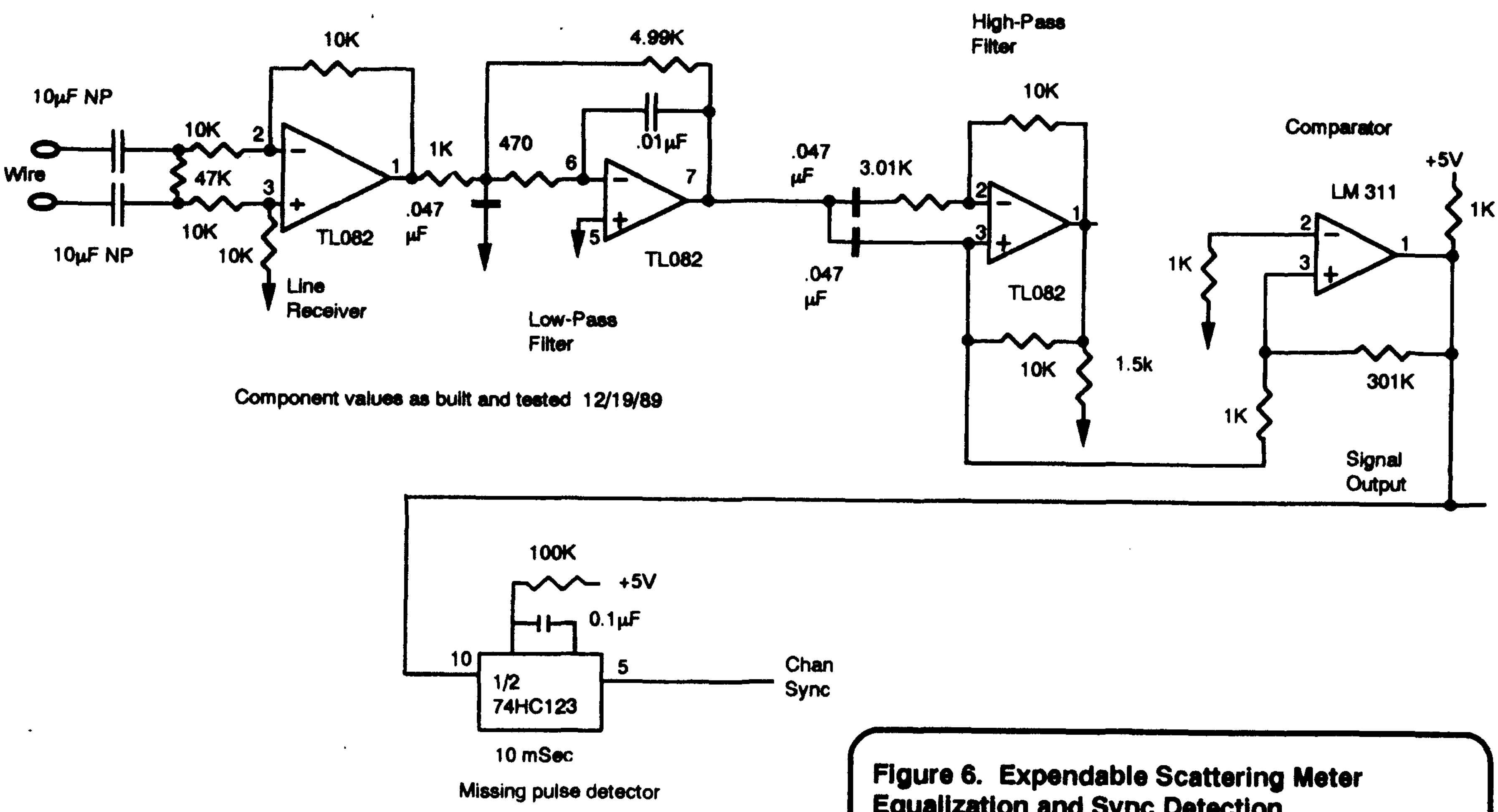


Figure 6. Expendable Scattering Meter Equalization and Sync Detection Copyright 1991 Sea Tech Inc.	
Drawn By: Mark Borgerson	**Date:** 5/29/91

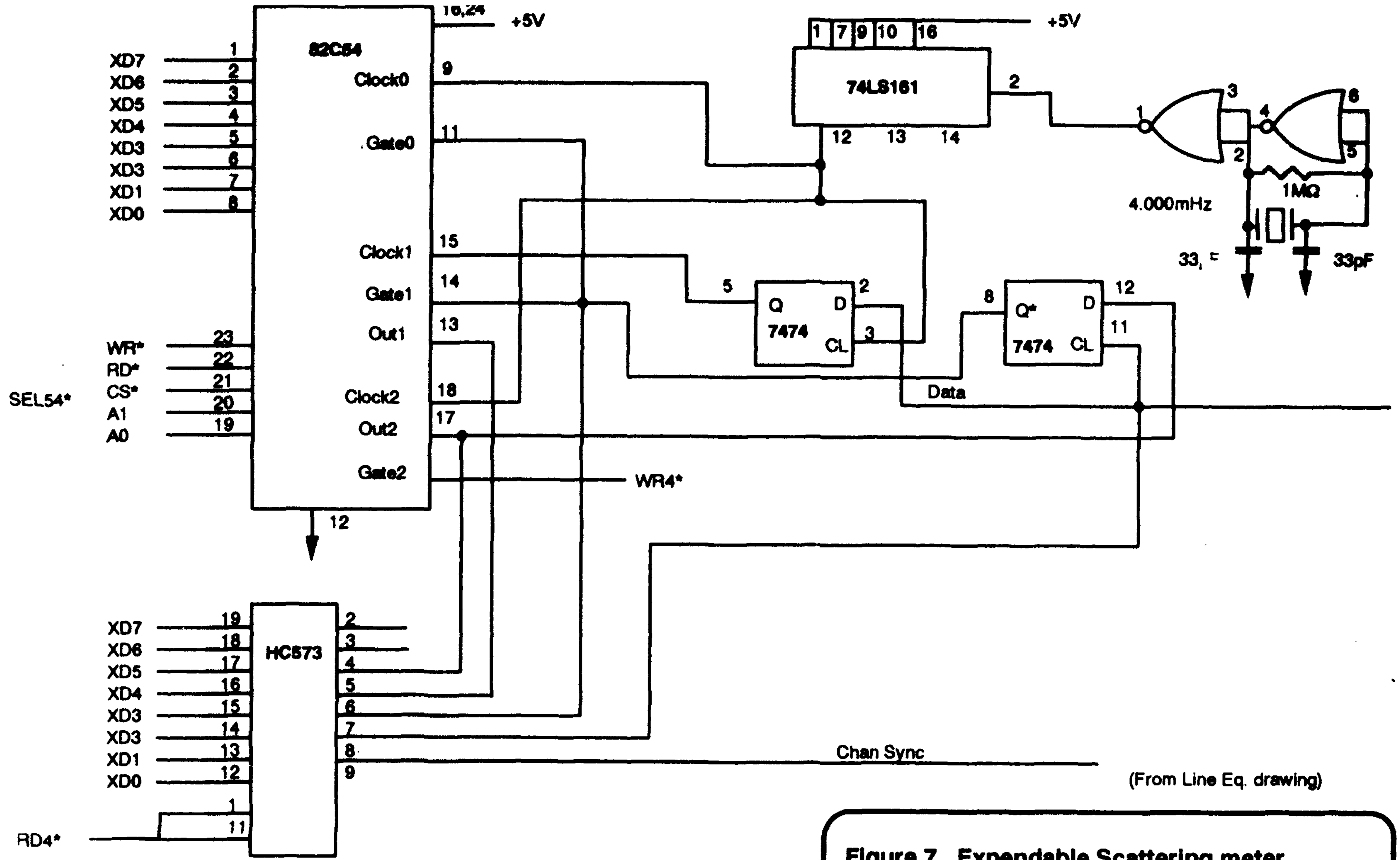


Figure 7. Expendable Scattering meter Period Counter

Drawn By:	**Date:**
Mark Borgerson	5/29/91

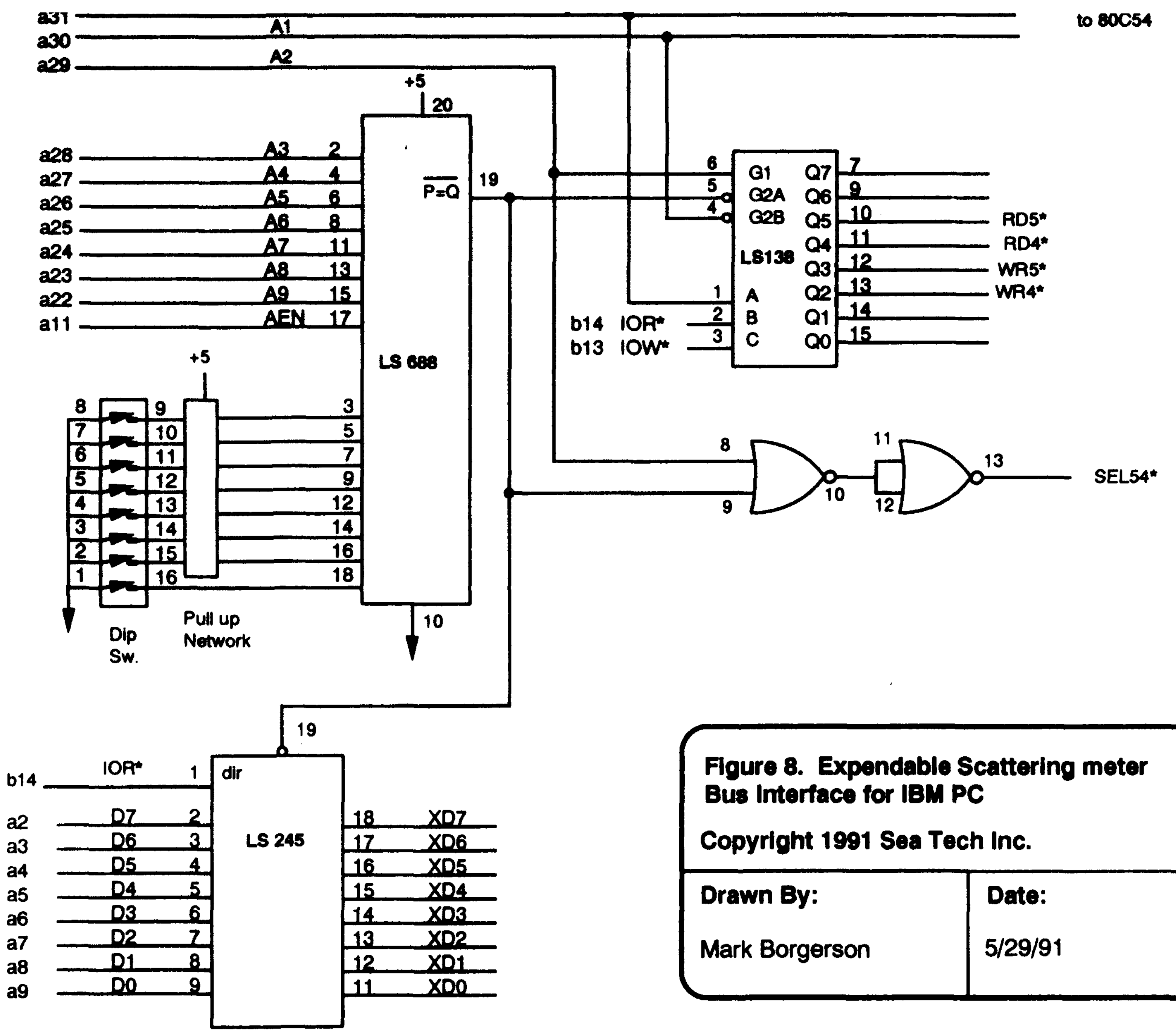


Figure 8. Expendable Scattering meter Bus Interface for IBM PC

Drawn By:	Date:
Mark Borgerson	5/29/91

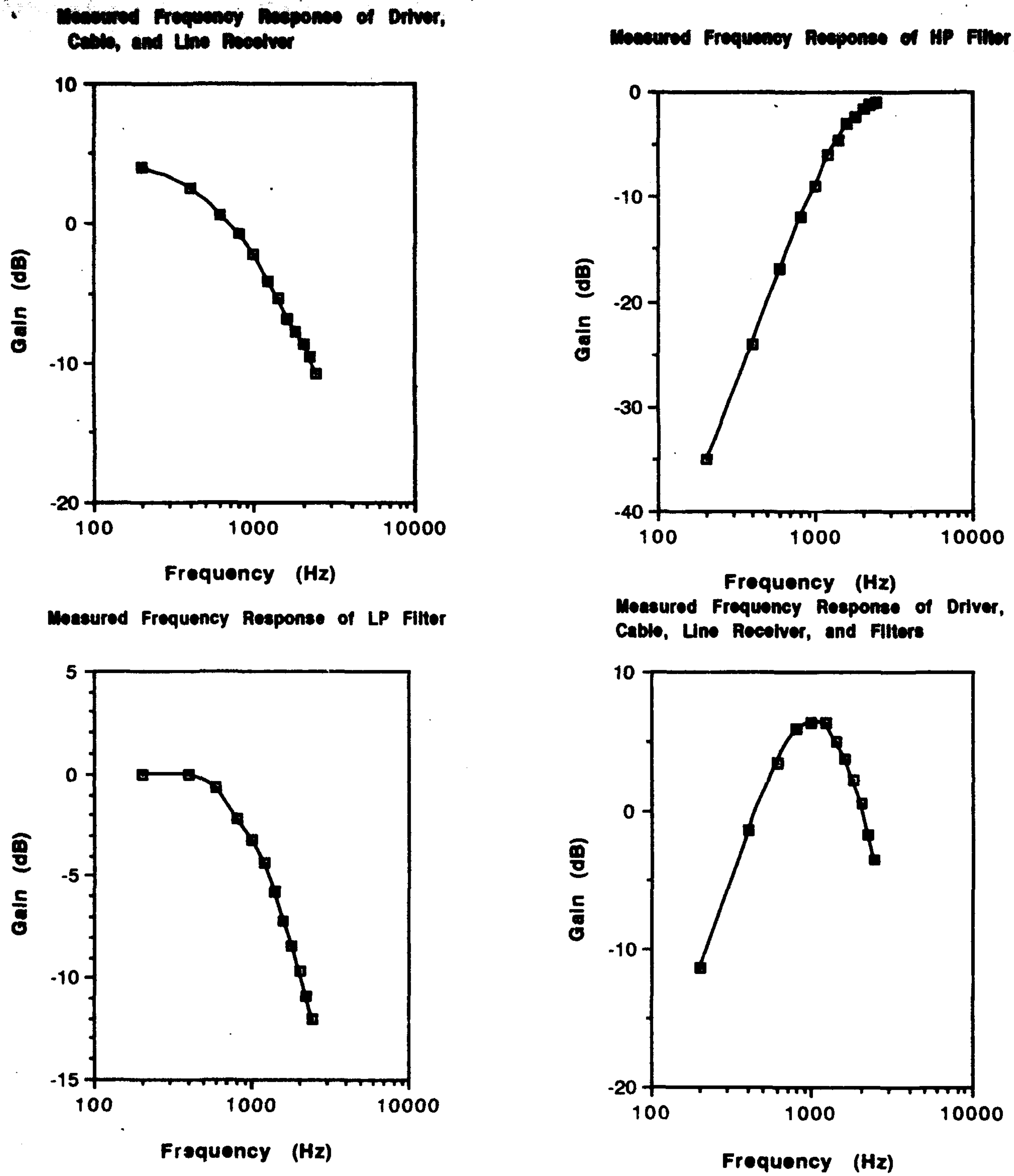


Figure 9. Frequency Response of Equalization Network

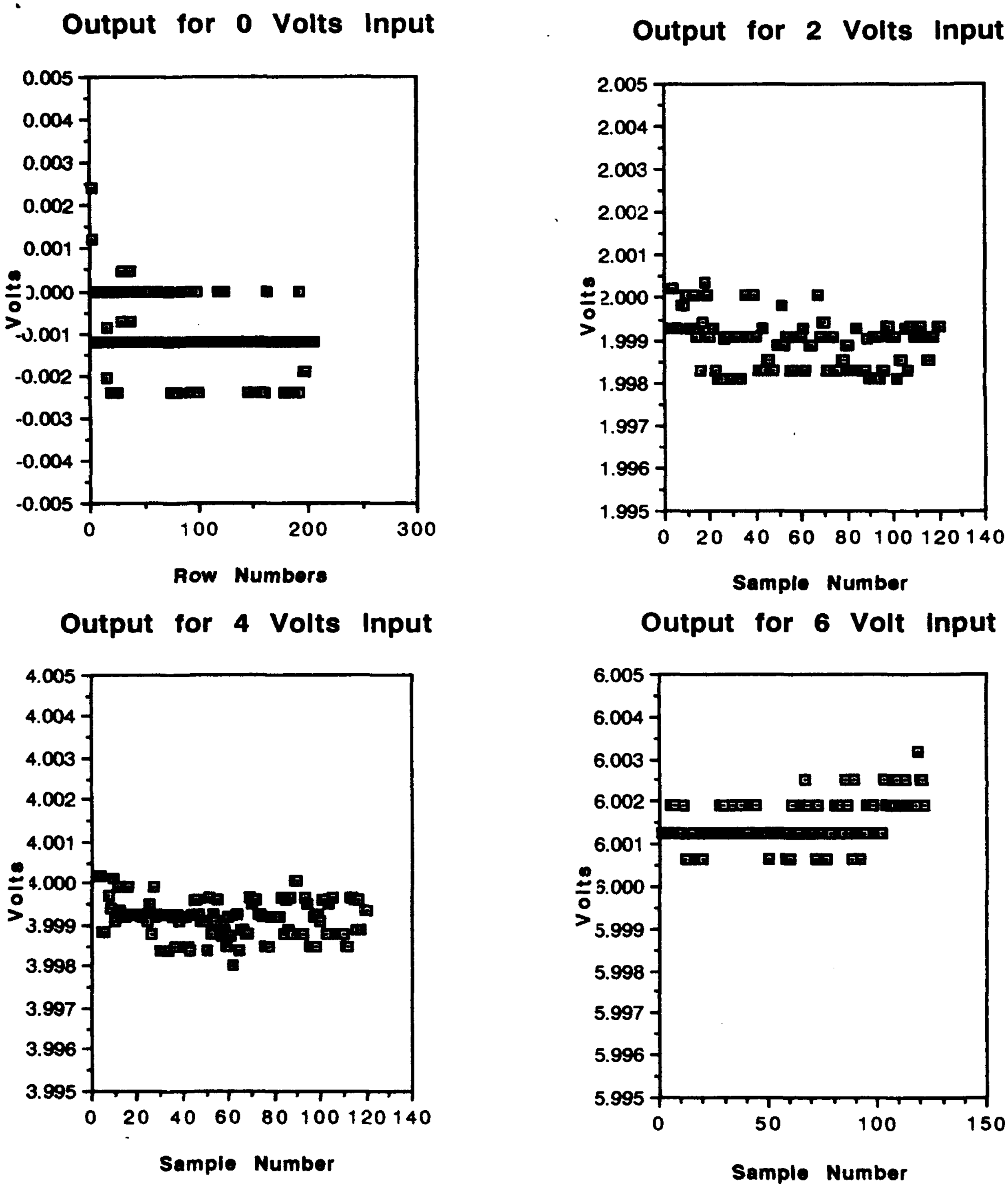


Figure 10. Received Voltages through Telemetry System for 0 to 6 Volts Input

Appendix A

Detailed Description of Sensor for Expendable or Non-Expendable Probe

The light scattering sensor is shown in Figure 1. This sensor consists of a light emitting diode (LED) source **1** modulated at approximately 750 Hz and driven with 50 milliamps of current resulting in a power output of approximately 10 milliwatts. The light source is encapsulated in an optically clear epoxy **9**. The light projects through the epoxy medium into the water towards the light detector **3**. The light is refracted at the water-epoxy interface **4**. Due to refraction the light ray ranges from approximately 90° at the top point **5** of the sample volume of water to approximately 12° at the lower point **6** of the sample volume **7**. The scattered light from particulate matter in the sample volume is measured by the solar blind light detector **3** which, like the light source, is encapsulated in epoxy **9**. The light detector water-epoxy interface refracts the scattered rays of light from the sample volume toward the detector **8**. The light stop **2** blocks direct light from the light source so the detector only measures the forward scattered light from the sample volume **7**. The light emitting diode **1** used is an infrared source emitting at a wavelength of 880 nm. In the improved version of the scattering sensor two LED's are connected in series and mounted side by side to double the light output of the light source. The light detector **3** is a silicon detector equipped with a light filter designed to block visible radiation; this prevents detector saturation from ambient light. The components are connected to a printed circuit board **10**. This simple and adaptable sensor module can be packaged in either a non-expendable probe or the Sparton expendable probe. The radius of the face of the module need only be the same as that of the probe in which it is housed.

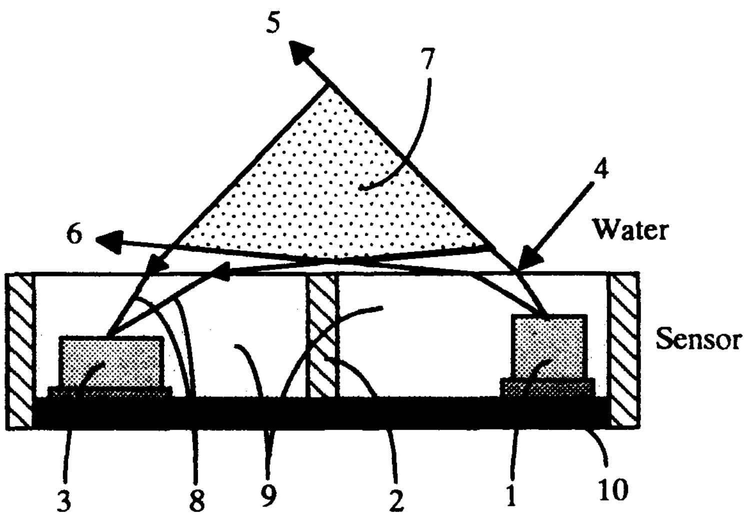


Figure 1 Underwater Forward Light Scattering Sensor

Appendix B

Performance of Forward Light Scattering Sensors in High Concentrations of Suspended Sediment

We tested three forward light scattering sensors with different pathlengths to evaluate their response in high concentrations of suspended solids. The optical design for all three sensors are similar to that shown in Figure 1.

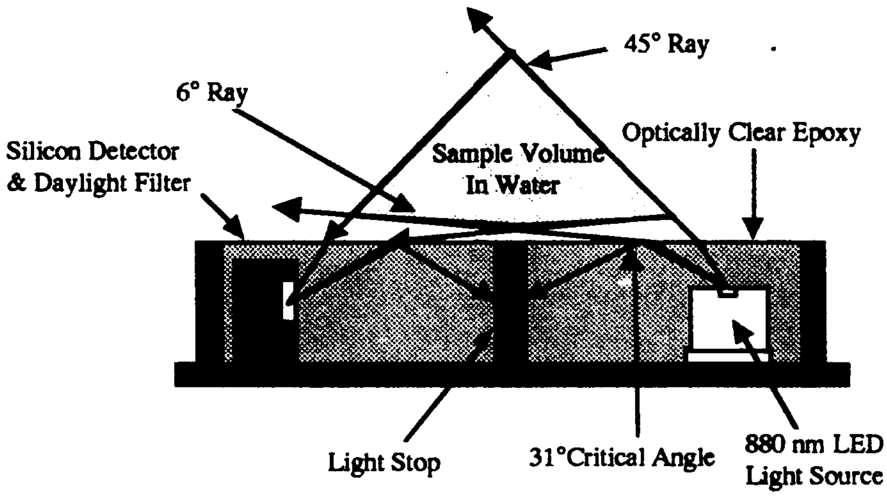


Figure 1. Forward Scattering Sensor Optical Diagram

An 880 nm LED and a detector are potted in clear epoxy; a light stop prevents light from the LED from reaching the detector directly through the epoxy. The epoxy and water interface refracts the LED light in the forward direction. The LED light is scattered by the particles in the sample volume and some of the scattered light reaches the detector. The amount of scattered light received by the detector is proportional to the concentration of suspended material in the water until the suspended material in water reaches very high concentrations. The range over which the sensor operates linearly depends on the distance traveled through water by the detected light.

Since a given ray which reaches the detector can be scattered from a particle anywhere in the sample volume, a range of pathlengths are possible. In this report, we characterize the forward light scattering sensors by the minimum pathlength a ray of light from the LED can travel and reach the detector. If total internal reflection can occur (as shown in Figure 1), then the minimum pathlength is the distance between the two points along the water and epoxy interface where total internal reflection occurs. If the light stop prevents total internal reflection, then the minimum pathlength is the width of the light stop multiplied by the cosine of the angle of a ray incident on the edge of the light stop.

We calibrated the three forward light scattering sensors with minimum pathlengths of approximately 2.5 millimeters, 7.5 millimeters, and 25 millimeters, and a 5 centimeter transmissometer with modeling clay. To avoid saturation at the large concentration levels, it was necessary to reduce the gain of the 2.5 and 7.5 millimeter minimum pathlength sensors by a factor of 500 from the gain used for open ocean profiling. The gain of the 25 millimeter minimum pathlength sensor was reduced by a factor of 100. Figure 2 is the calibration of the light scattering sensors; Figure 3 is a calibration of the transmissometer.

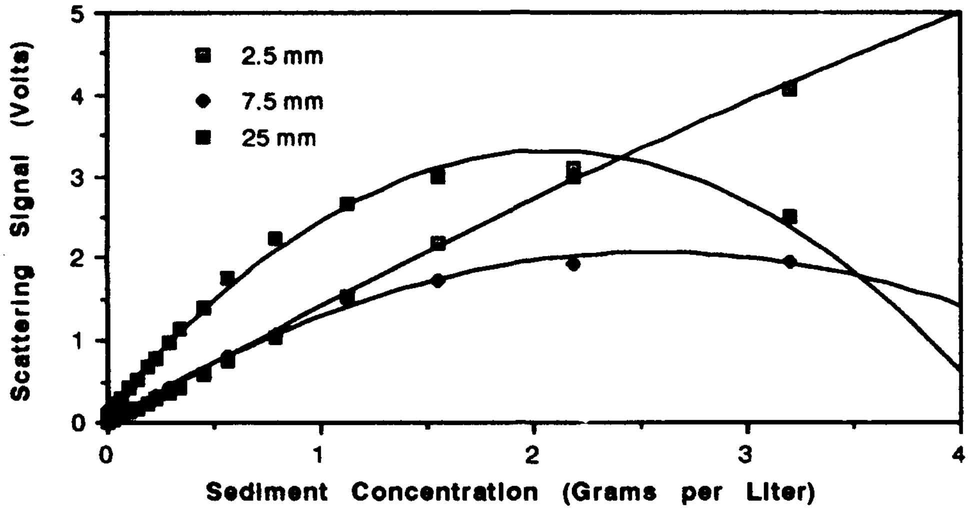


Figure 2. Calibration of Forward Light Scattering Sensors with Modeling Clay

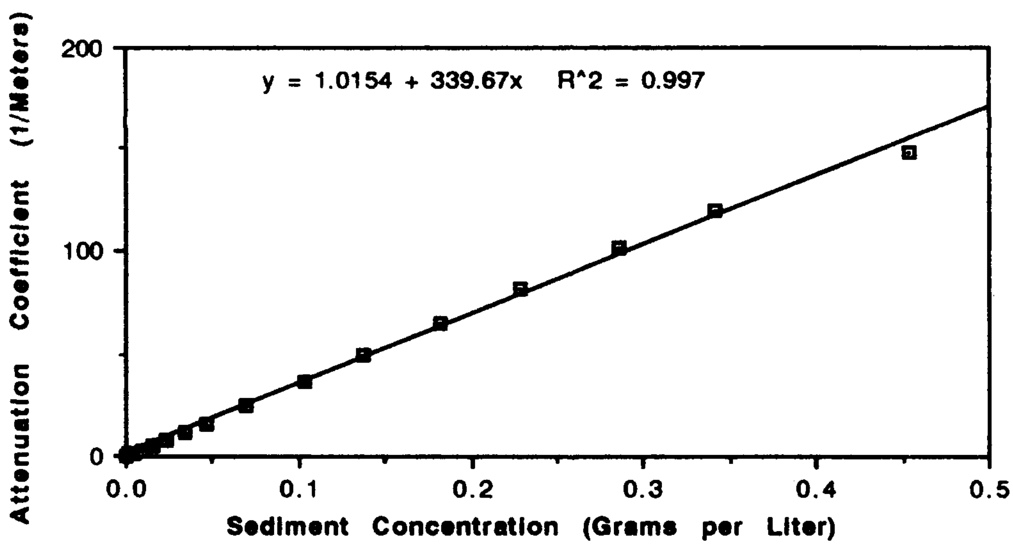


Figure 3. Calibration of a 5 Cen ter Transmissometer with Modeling Clay

The calibrations show that the forward light scattering sensors were all able to measure the clay concentrations linearly to above .5 grams per liter, which is beyond the limits of the transmissometer. At .45 grams per liter, the transmissometer output was only one millivolt. The scattering sensor with a 2.5 millimeter minimum pathlength was linear up to the highest sediment concentrations against which it was calibrated -- over 3 grams per liter. The scattering sensor with a 7.5 millimeter pathlength operated linearly to approximately 1 gram per liter, and the scattering sensor with a 25 millimeter minimum pathlength was linear to about .5 grams per liter.

We can produce curves similar to those in Figure 2 using a simple model. This model assumes that absorption is the parameter which limits the linearity of the scattering sensor. When sediment concentrations become high enough that a significant portion of the light scattered in the direction of the detector is absorbed, the output of the scattering sensor is no longer linear with increasing concentrations. We determined the relationship between concentration of the clay and the attenuation coefficient, c, from low concentration data points from the calibration of the transmissometer:

$$c = 0.34 + 320*conc$$

We estimate the absorption coefficient, a, due to particles in the water to be an order of magnitude less than the attenuation coefficient due to particles in the water:

$$a = 0.34 + 32*conc$$

We then assume that the scattered light received by the detector is some constant multiple of the sediment concentration multiplied by a loss factor due to absorption. Therefore:

$$V = k*conc*exp(-z*a)$$

where V is the sensor output, k is some constant, and z is the minimum pathlength of the sensor. Using these equations, we generated a curve for output versus sediment concentration for each of the three sensors. The plot is shown in Figure 4.

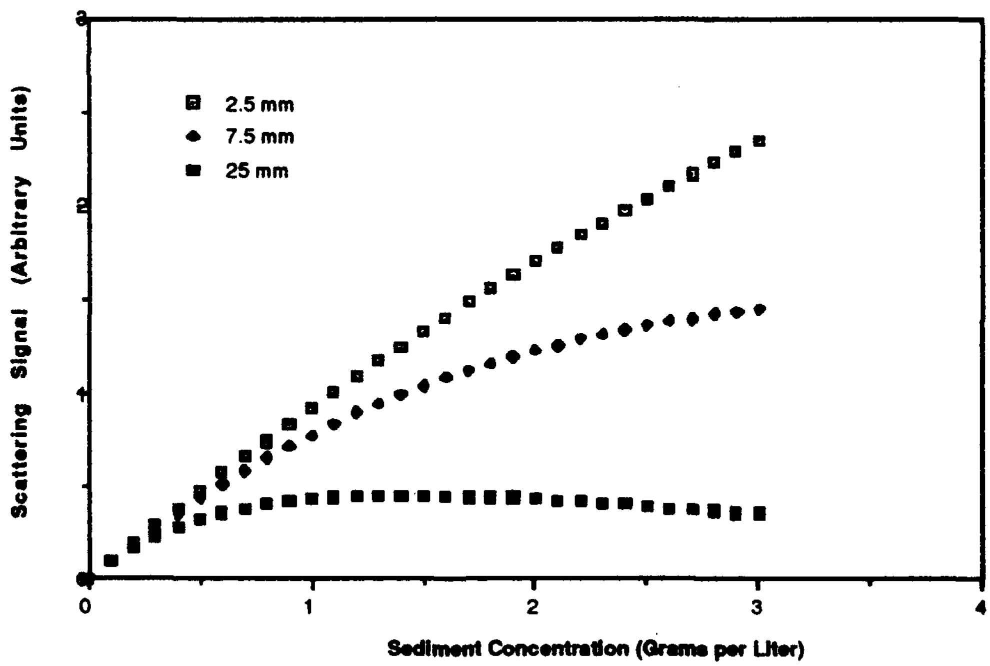


Figure 4. Theoretical Sensor Output Based on a Simple Model

The shapes of the curves are similar to those in Figure 2. Therefore, it appears that this model may describe the situation fairly well. Calibrating the sensors along with an absorption meter, so that the absorption coefficient was measured, would allow us to better assess this model.

Appendix C

```
program Newexp;
program to test interface to expendable scattering meter}
{updated 7/17/91  MJB}
Uses DOS, CRT,
const
    iobase = $340;
    ctlReg = iobase+3;
    statByte = iobase+4;

    maxftime = 150;

type grpArray = array[0..3,1..3] of real;

var ch: char;
   bt:byte;
   i,j:word;
   infreqs, volts:grparray;
   ctimes:array[1..3] of integer;
   counttime, chanpd:integer;
   framesum:longint;
        c,d,fscale,f0: real;
Procedure SetGateTime(counts:word);
Begin
        port[ctlreg]:= $B2; {sets mode 1}
        Delay(3);
        Port[iobase+2]:= counts and 255;
        port[iobase+2]:= (counts div 255);
End;

Function Freq: Real;
var th,tl:byte;
        tcount,incount:word;
        ctime:real;

Begin
    port[ctlreg]:= $30;{set time counter 0 for mode 0}
    Delay(1);
    port[iobase]:= 0; {set initial count to 0000}
    port[iobase]:= 0;
   {now set up the input counter}
   port[ctlreg]:= $70; {counter 1 to mode 0}
   Delay(1);
   port[iobase+1]:= $0;
   port[iobase+1]:= $0;

        {now open the gate for preset time}
        port[iobase+4]:=0; {trigger the count}
        Delay(10); {wait for gate to open}
        repeat until (port[statbyte] and 8)=0; {wait for gate to close}
        tl:= port[iobase];
        th:= port[iobase]; {read the count--gate is closed}
        tcount:= 65536-(th*256+tl);
        ctime:= 0.4999*tcount;
```

```
      tl:= port[iobase+1]; {read low count from input}
  th:= port[iobase+1];
  incount:= 65536-(th*256+tl);

  If ctime > 0 then
  freq:= incount/ctime*1e6
  else freq := 5000;
  counttime:= trunc((ctime+500)/1000);
End;

Procedure FrameSync;
{sync goes low after 10 msec of no input pulses from wire}
var stime:integer;
Begin
      stime:= 0;
      Repeat
      Delay(1);
        inc(stime);
      until ((port[statbyte] and 2)=0) or (stime >1000); {wait for sync low}
      Repeat
            inc(stime);
            {Delay(1);}
      until ((port[statbyte] and 2)=2) or (stime > 1000); {wait for sync high}
end;
Function FrameInterval:integer;
var ftime: integer;

Begin
  ftime:= 10;
  FrameSync; {wait for first sync pulse}
  Delay(ftime); {can't handle frame time less than 10 msec anyway}
  Repeat
            inc(ftime);
            Delay(1);
      until ((port[statbyte] and 2)=0) or (ftime > maxftime); {wait for sync low}
      Repeat
            inc(ftime);
            Delay(1);
      until ((port[statbyte] and 2)=2) or (ftime > maxftime); {wait for sync high}
      FrameInterval:= ftime;
End;

Procedure ReadFreqs;
var i,group, dtime: integer;
Begin
      for group:= 0 to 1 do
      Begin
            FrameSync;
            Delay(1);
      for i:= 1 to 3 do
      Begin
      Delay(1);
      infreqs[group,i]:= freq;
      ctimes[i]:= counttime;
```

```
                    Dtime :=chanpd-counttime-4;
                    If dtime > 0 then delay(dtime);
            End;
        end;
    End;

    Procedure WriteGroup(gidx:integer);
    var i:integer;
    Begin
            for i:= 1 to 3 do
            Begin
                    Write(infreqs[gidx,i]:8:2);
            End;
    End;

    Procedure WriteVGroup(gidx:integer);
    var i:integer;
       v,f: real;
    Begin
            for i:= 1 to 3 do
            Begin
              f:= infreqs[gidx,i];
                            v:=(f-f0)/fscale;
                    Write(v:8:3);
            End;
    End;

    Procedure WriteResults;
    var i,j, grpidx, outidx:integer;
       f5:real;
    Begin
          {find sync phasing so gnd is in freq 3 of first group
           and vcc is in freq 3 of third group }
          grpidx:= 0;
                    for i := 0 to 1 do
                    begin
                      j:= (i+1) and 1; {rolls over above 1}
                      if (infreqs[i,3]>1000) and (infreqs[j,3]>800) then grpidx:= i
                    end;
                    f0:= infreqs[grpidx,3];
                    f5:= infreqs[(grpidx+1) and 1,3];
                    Gotoxy(1,24);
                    Clreol;
                    Write(f0:8:2, f5:8:2,fscale:8:2);
                    fscale:=(f5-f0)/6.2;
            Gotoxy(1,5);
          outidx:= grpidx;
          WriteGroup(outidx);
            Gotoxy(31,5);
          outidx:= (grpidx+1) and 3;
          WriteGroup(outidx);

                    gotoxy(66,6);
                    for i:= 1 to 3 do write(ctimes[i]:4);
```

```
      Gotoxy(1,8);
    outidx:= grpidx;
    WritevGroup(outidx);
      Gotoxy(31,8);
    outidx:= (grpidx+1) and 3;
    WritevGroup(outidx);

end;

Procedure DoFrameCheck;
Begin
  ClrScr;
  Writeln('Checking for proper frame sync pulses.');
  framesum:= (frameinterval+FrameInterval+frameInterval) div 3;
  If (frameInterval < 60) or (frameInterval > 120) then
  Begin
      Writeln('Bad Frame Interval   (',frameinterval,')');
    framesum:= 80; {default value}
  End;

  Write('Frame interval = ', framesum);
  chanpd:= framesum div 4;
  Writeln('    Channel Period = ',chanpd);

End;
Procedure DoGetData;
var ch: char;
Begin
    Write('Press any key to end data display.');
      repeat
            ReadFreqs;
          WriteResults;
      until keypressed;
    ch:= readkey;
End;
Procedure DoSetGate;
var dcint:real;
Begin
  Write('Data Collection interval in milliseconds: ');
  Readln(dcint);
  Setgatetime(trunc(dcint*2000)); {for 500 khz clock}
End;

Procedure ShowMenu;
Begin
      Gotoxy(1,16);
   Writeln('Program Options:');
   Writeln('  F:  Frame Interval Check');
   Writeln('  D:  Data Display');
   Writeln('  G:  Set Gate Interval');
   Writeln('  Q:  Quit');
   Write('Select one by letter: ')
```

```
End;


Begin
      SetGateTime(6500);
    DoFrameCheck;
    Repeat
       ch:= ' ';
       ShowMenu;
       ch:= upcase(Readkey);
       Gotoxy(1,23);
       ClrEol;
       Case ch of
       'F': DoFrameCheck;
       'D': DoGetData;
       'G': DoSetGate;
       End;
    until ch= 'Q'
End.
```